APPLICATION

FOR

UNITED STATES LETTERS PATENT

TITLE: PARTICLE CONTROL USING A PATH

APPLICANT: THOMAS M. CRONIN

CERTIFICATE OF MAILING BY EXPRESS MAIL

Express Mail Label No. EL688320839US

I hereby certify that this correspondence is being deposited with the United States Postal Service as Express Mail Post Office to Addressee with sufficient postage on the date indicated below and is addressed to the Commissioner for Patents, Washington, D.C. 20231.

October 26, 2001
Date of Deposit

Signature

Mike Augustine
Typed or Printed Name of Person Signing Certificate

PARTICLE CONTROL USING A PATH

TECHNICAL FIELD

This invention relates to modeling and controlling particles using a path.

BACKGROUND

Particle systems are often used to model animations of "flowing" systems, that is, modeling a set of objects that possess some common characteristics and/or movements. For example, particle systems are sometimes used to model water falling, smoke rising from a chimney, a flock of birds flying, snow falling, schools of fish swimming, etc.

A particle system is essentially a set of rules that are used to define how to "generate" and "render" particles for display on a display device such as a computer display screen. Generating particles includes defining particle attributes for each particle to be modeled. For example, particle attributes may include an age, a physical position, a velocity (speed and direction), a color and a size. Typically, following particle generation, a computer processor is used to render the particles by converting the particle attributes for each particle into display device information. The display device information is then used to output the

rendered particle information by controlling individual pixels on the display device. Individual particles may be rendered and displayed as dots, lines, polygons or more complex objects.

To create an animation using a particle system, particle "movements" are displayed in a series of frames where the particle attributes are up-dated, rendered and then displayed on the display device. Typically frame up-dates occur 50-60 times per second. Particle system animations may model thousands of individual particles. Therefore, to transmit a particle system animation over an electronic network from a first computer to a second computer requires a large amount of bandwidth. For example, to display a particle system animation on a client computer that is defined by a server computer on the Internet, the server computer must transmit large amounts of particle information to a client on a frame-by-frame basis.

DESCRIPTION OF DRAWINGS

FIG. 1 shows a block diagram of an electronic network having a server computer and a client computer;

FIGS. 2A and 2B shows abstract representations of a path construct defined by control points and connected by

a spline curve;

FIG. 3 is a flowchart showing a particle system animation process using a path construct; and

FIG. 4 is a block diagram of computer hardware on which the particle system animation process may be implemented.

DESCRIPTION

Referring to FIG. 1, an electronic network 10 that is used to transfer particle system information includes a server computer 12 (a "server") and a client computer 20 (a "client") that are both connected to send and receive data over a network link 18. Network link 18, may be operated as a "private" or a "public" network, for example, network link 18 could be a local area network, or, network link 18 could be a public network, such as an Internet 30. Server 12 is also connected to one or more storage devices 14 that contain files 16 (e.g., data and programs).

In electronic network 10, to model and display a particle animation on client 20, server 12 sends data that defines a particle "control" path over network link 18 to client 20. A path is a mathematically-defined continuous connection of a set of points, where each point is defined

by axial coordinates that correspond to coordinates on a display device. "Control" refers to an algorithm associated with the path that is used to change a particle's attribute during a frame up-date. For example, a control algorithm may change a particle attribute based on a determined distance between the particle and the closest point on the path to the particle during a frame up-date.

Client 20 uses the control path definition to change particle attributes for particles included in the particle animation in a series of up-dates. The changed particle attributes are then rendered and displayed as a series of frames on a display device. This way of modeling and displaying a particle animation on client 20 does not require server 12 to generate and up-date individual particle attributes for client 20. Instead, the generation, up-dating and rendering of particles is performed on client 20. Therefore, even a complex particle path that is being used to animate thousands of particles may be achieved by passing a small amount of information over network 18 that requires relatively little bandwidth. Furthermore, server 12 does not need to send the particle generation information for a set of particles, instead client 20 may generate an initial set

of particle attributes and then update and render the particles using the control path construct sent by server 12. Also, since client 20 performs the rendering, a user on client 20 can interact with the particle animation to make changes to the displayed screens without having to wait for communication of those changes between client 20 and server 12.

Referring to FIG. 2A, a representation of a particle control path 40 is shown as a continuous path 40 that has been interpolated between a series of points 42, 44, 46 and 48. Each point 42, 44, 46 and 48 is defined by axial coordinates that correspond to axial coordinates on the display device controlled by the client 20. In a particle system animation, in order to display particle movements that appear "smooth" and "natural", the connection of the points included in the path must also be done smoothly. One way of connecting points smoothly is with a "spline curve", i.e. a mathematically-defined algorithm that is used to interpolate curves between points that avoids the sharp corners that might occur if connecting directly from point-to-point with straight lines. Not all spline curves actually connect with each defined point. In one embodiment, the points are connected with a "Catmull-Rom" spline curve, which is an interpolated curve that passes

through all points. It is noted that any mathematical construct that enables an interpolation of a line between points could be used.

Referring to FIGS. 2A and 2B, a representation of control path 40, that connects points 42, 44, 46 and 48, is generated using software. To illustrate the use of control path 40 in a particle animation, an exemplary particle X is shown (See FIG. 2B) being influenced by control path 40 in a series of eleven (11) frame up-dates 0-10. In this example, particle X's attributes include an initial position, velocity (speed and direction) and an "age". In this example, particle X's age begins at age zero, and the age increases by one during each frame up-date. During each frame up-date 0-10, an "attractor point" is determined for particle X, i.e., the closest point on control path 40 to particle X. In operation, during up-date frame 1, point P1 is the determined attractor point, during up-date frame 3, point P3 is the determined attractor point, and so forth. The determined attractor point is then used to determine the distance between particle X and the current attractor point. In this example, the control algorithm associated with path 40 is defined to "attract" particle X towards the determined attractor point. The amount of "attractive

force" towards path 40 is based on the determined distance between the particle and the attractor point. The determined attractive force (or alternatively, a repulsive force) will cause a change to one or more of the particle X's attributes, e.g., physical position, speed and/or velocity of the particle. It is noted that particle X is generally closest to an attractor point that is between two of the set of points 42, 44, 46 and 48.

Continuing with this example, after particle X's attributes are up-dated, particle X is rendered and displayed on a client 20 display device. The resulting animated "movement" of particle X from frame-to-frame is a relatively smooth "flow", that is, with some variation in the movement of particle X according to its age, and proximity to different attractor points on control path 40. As stated previously, generally, during a frame up-date, the attractor point on control path 40 will be between two of the set of points 42, 44, 46 and 48, therefore, the location of the attractor point will be interpolated from the mathematically-defined path construct used to connect the points 42, 44, 46 and 48.

The control algorithm that is used to define a the change to a particle attribute may vary. Typically, an algorithm includes a set of variable definitions, for

example:

'd' (A distance from the closest point on the path to the particle being up-dated);

'c' (A constant value that may be set by the programmer or a user); and

'f' (An amount of attractive force, i.e., the amount of force to applied from the current position of the particle towards the attractor point on the path).

Using these variable definitions, exemplary control algorithms could be defined as follows:

1) f = d x c; This algorithm enables the application of a force that is determined by the distance multiplied by the constant;

2) f = d x d x c; This algorithm enables the application of a force that is determined by the distance squared(d x d); and

3) f = c; This algorithm enables the application of a force that is determined from, and equal to, the constant

(c).

Other control algorithms could be used, for example, control algorithms that include more variables, e.g.:

'r'; A radius (a distance) from the closest point to the particle where no force will be applied.

As an example of a control algorithm that uses the variable 'r':

4) f = (d-r) x c; This algorithm enables the application of a force to a particle only when the particle is outside of the radius distance, 'r'.

It is noted that other particle attributes besides a particle's age could be used to determine when and how to influence a particle attribute with a path construct, for example, a particle's velocity or color could be used. Also, it is noted that the path curve representations shown in FIGS. 2A and 2B are not, generally, rendered or displayed as part of the particle animation, instead,

FIGS. 2A and 2B are abstract representations of the continuous path connection interpolated between a set of points.

Referring to FIG. 3, a process 100 is shown for performing a particle animation. Process 100 includes sending 110 a particle path from a server to a client, receiving 120 the particle path at the client, generating 130 an initial set of particle attributes, rendering and outputting 140 particles on the client display device, up-dating 150 particle attributes during a frame up-date, determining 160 each particle's age, determining 170 a closest point on path to the particle being up-dated, up-dating 180 the particle's attributes according to the determined closest point on the path, and, determining 190 whether all particles in the animation have been up-dated. If all particles have not been up-dated, method 100 repeats the sequence of actions 160, 170 and 180 and 190. If all particles have been up-dated, method 100 returns to rendering and outputting 140 the up-dated frame on the client display device.

Fig. 4 shows a personal computer 200 on which process 100 may be implemented. Computer 200 includes a processor 210, a memory 220, and a storage medium 230. Storage medium 230 stores data for one or more particle system

animations and machine-executable instructions 240 that are executed by processor 210 out of memory 220 to perform process 100.

Although a personal computer 200 is shown in Fig. 4, process 100 is not limited to use with the hardware and software of Fig. 4. It may find applicability in any computing or processing environment. Process 100 may be implemented in hardware, software, or a combination of the two. Process 100 may be implemented in computer programs executing on programmable computers or other machines that each include a processor, a storage medium readable by the processor (including volatile and non-volatile memory and/or storage components), at least one input device, and one or more output devices. Program code may be applied to data entered using an input device (e.g., a mouse or keyboard) to perform process 100 and to generate output information.

Each such program may be implemented in a high level procedural or object-oriented programming language to communicate with a computer system. However, the programs can be implemented in assembly or machine language. The language may be a compiled or an interpreted language.

Each computer program may be may be implemented as one or more articles of manufacture (storage media), such

as a CD-ROM, hard disk, or magnetic diskette, that is readable by a general or special purpose programmable computer for configuring and operating the computer when the storage medium or device is read by the computer to perform process 100. Process 100 may also be implemented as a machine-readable storage medium, configured with a computer program, where, upon execution, instructions in the computer program cause a machine to operate in accordance with process 100.

Particle animations are generally used to animate only a portion of a computer display device and are commonly rendered in a "third-dimension" ("3D"). More specifically, an initial two-dimensional ("2D") background is rendered and displayed, then other "3D" objects are rendered and displayed in front of the 2D background display. As an example, an initial 2D background display might include a mountain and a night sky above, then a house with a chimney is rendered and displayed in "3D", that is, in front of the mountain and sky 2D background. A particle system might be specified in 3D space and used to animate smoke that rises from the chimney and into the night sky. The particle path may then be used to animate the movements of the generated smoke particles away from the chimney peak and rising towards the sky. It is noted

that particle animations may also be specified, rendered and displayed in "2D".

Though specific embodiments have been described other ways to implement the features of those embodiments are possible. For example, multiple paths may be defined for a particle animation, where different paths are defined by different sets of control points and, therefore, may cause different degrees or amounts of change to a particle's attributes.

A number of embodiments of the invention have been described. Nevertheless, it will be understood that various modifications may be made without departing from the spirit and scope of the invention. Accordingly, other embodiments are within the scope of the following claims.